**Hawk Eyes**

1. Introduction

The goal of this project is to setup a self-service data collection platform. For many research/application area, collection data is usually the first step, and in most time one of the most important step. To avoid user to write their own script to crawl data every time, Hawk Eyes will allow user to customize his collection job template, and host the crawler task on Hawk Eyes platform. User also could view/manage his jobs in self-service mode.

1. Spec.

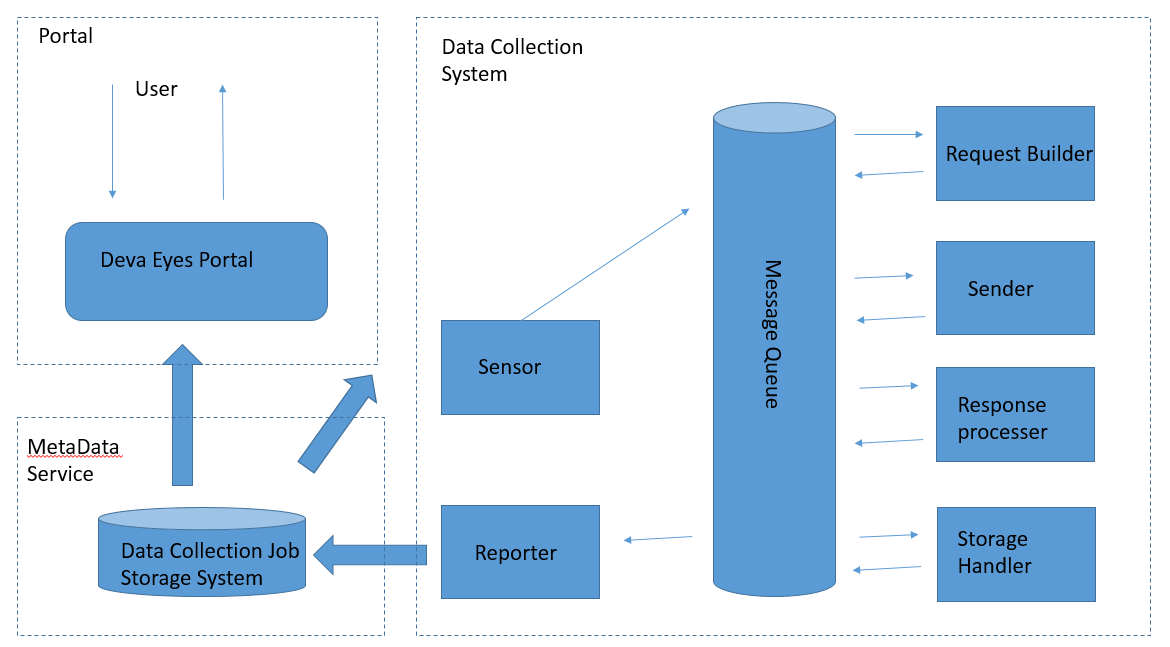
The feature Hawk Eyes will support in V1:

1. Crawler data with text/xml/json format
2. Crawler with http protocol. And support crawler with specified cookie/http header.
3. Post data processing, extract data from http response, according to user’s configuration.
4. Periodically job, which will be auto-scheduled every day/hour, auto detect is there any update in data source.
5. Flexible storage option. HawkEyes will allow user to store result in file/data base/any other cloud service.
6. Self-service management of data collection tasks from a Web portal.
7. Horizontal scalability.

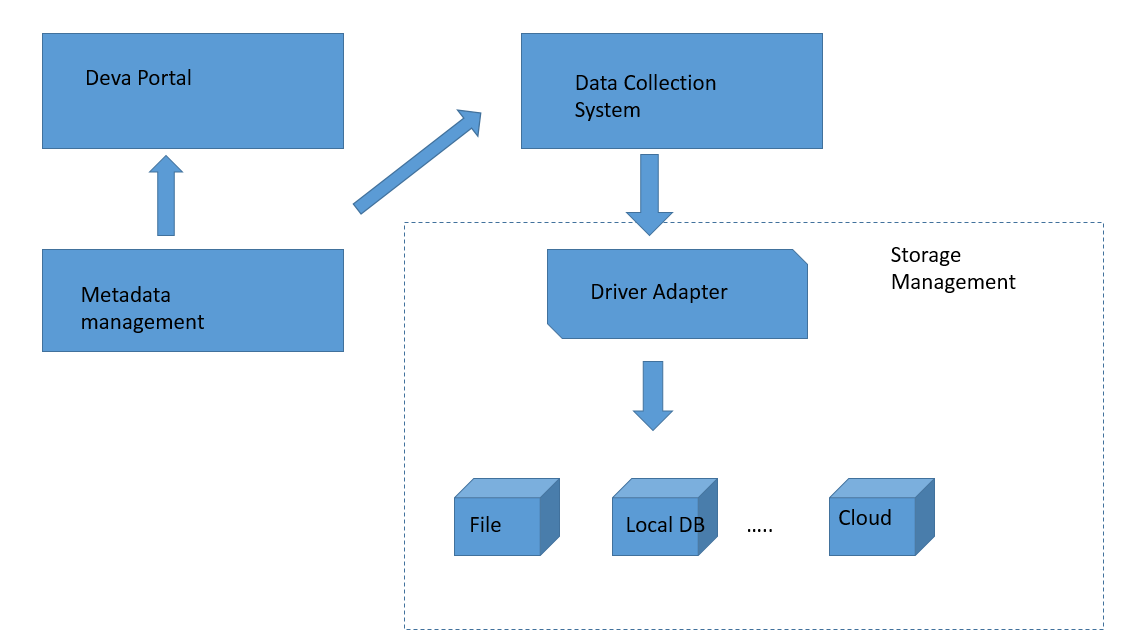
Features will not support in V1:

1. Crawler for img/video data.
2. Crawler with ftp/https protocol.
3. Quota management / Throughput control. Performance is not the main target in V1.
4. Async crawler mode.
5. View data collection result.
6. Cancel task.
7. Architecture

For data collection.



For data storage



1. Mile Stone

M1: Setup MQ and basic crawler functionality. User could submit crawler job directly to MQ to get http response.

1. Define the Message and MQ interface. Investigate on existing MQ solution, chose one of them as our implementation.
2. Implement worker node interface and framework. Implement framework infrastructure component, like Log.
3. Implement basic crawler component, like request builder/sender/response processor.
4. Integration test.

M2: Finish the job template (XML format) and the response process template (extract data from http response). User could define complicate data collection jobs like crawler data with specified url pattern under a specified domain.

1. Design the job template formatting, and the template parser.
2. Design the response process template.
3. Implement template parser and response processor.
4. Integration test.

M3: Setup meta data service, user could submit job to meta data service, which will be automatically pick up by Hawk Eyes. Setup the Storage component, user could specify where to store the collected data.

1. DB setup.
2. Design DB schema (table structure)
3. Implement a service that could pull/insert data from metadata DB.
4. Implement Sensor and Reporter.

M4: integrated with Web portal.

1. Choose a web portal solution (java/php/nodejs)
2. Implement basic web UI based on metadata service.
3. Integration test.